# CS 5350/6350: Machine Learining Spring 2018

Homework 2

Handed out: 19 January, 2018
Due date: 5 March, 2018

## General Instructions

- You are welcome to talk to other members of the class about the homework. I am more concerned that you understand the underlying concepts. However, you should write down your own solution. Please keep the class collaboration policy in mind.
- Feel free discuss the homework with the instructor or the TAs.
- Your written solutions should be brief and clear. You need to show your work, not just the final answer, but you do *not* need to write it in gory detail. Your assignment should be **no more than 10 pages**. Every extra page will cost a point.
- Handwritten solutions will not be accepted.
- The homework is due by **midnight of the due date**. Please submit the homework on Canvas.
- Some questions are marked **For 6350 students**. Students who are registered for CS 6350 should do these questions. Of course, if you are registered for CS 5350, you are welcome to do the question too, but you will not get any credit for it.

## 1 Expressiveness of Linear Classifiers

1. [60 points] Can you figure out an equivalent linear classifier for the following boolean functions? Please point out what the weight vector is, what the bias parameter is, and what the hyperplane is. Note that the hyperplane is determined by an equality. If you cannot find out such a linear classifier, please explain why.

   (a) $f(x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3$

   (b) $f(x_1, x_2, x_3) = x_1 \wedge \neg x_2 \wedge \neg x_3$

   (c) $f(x_1, x_2, x_3) = \neg x_1 \vee \neg x_2 \vee \neg x_3$

   (d) $f(x_1, x_2, \ldots, x_n) = x_1 \vee x_2 \ldots \vee x_k$ (note that $k < n$).

   (e) $f(x_1, x_2, x_3, x_4) = (x_1 \vee x_2) \wedge (x_3 \vee x_4)$

   (f) $f(x_1, x_2, x_3, x_4) = (x_1 \wedge x_2) \vee (x_3 \wedge x_4)$

2. [50 points] Can you draw equivalent decision trees for the following boolean functions? Note that you do NOT need to run the ID3 algorithm to learn such a tree. You only need to brainstorm and draw one. If you cannot, please explain why.

(a) $f(x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3$

(b) $f(x_1, x_2, x_3) = x_1 \wedge \neg x_2 \wedge \neg x_3$

(c) $f(x_1, x_2, x_3) = \neg x_1 \vee \neg x_2 \vee \neg x_3$

(d) $f(x_1, x_2, x_3, x_4) = (x_1 \vee x_2) \wedge (x_3 \vee x_4)$

(e) $f(x_1, x_2, x_3, x_4) = (x_1 \wedge x_2) \vee (x_3 \wedge x_4)$

3. [10 points] What do you conclude about the expressiveness of decision trees and linear classifiers from Problem 1 and 2? Why?

4. [30 points] The following boolean functions cannot be represented by linear classifiers. Can you work out some feature mapping such that, after mapping all the inputs of these functions into a higher dimensional space, you can easily identify a hyperplane that separates the inputs with different corresponding boolean function values? Please write down the separating hyperplane as well.

(a) $f(x_1, x_2) = (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$

(b) $f(x_1, x_2) = (x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_2)$

(c) $f(x_1, x_2, x_3)$ is listed in the following table

| $x_1$ | $x_2$ | $x_3$ | $f(x_1, x_2, x_3)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |

5. [40 points] [**For 6350 students**] Given two vectors $\mathbf{x} = [x_1, x_2]$ and $\mathbf{y} = [y_1, y_2]$, find a feature mapping $\phi(\cdot)$ for each of the following functions, such that the function is equal to the inner product between the mapped feature vectors, $\phi(\mathbf{x})$ and $\phi(\mathbf{y})$. For example, $(\mathbf{x}^\top \mathbf{y})^0 = \phi(\mathbf{x})^\top \phi(\mathbf{y})$ where $\phi(\mathbf{x}) = [1]$ and $\phi(\mathbf{y}) = [1]$; $(\mathbf{x}^\top \mathbf{y})^1 = \phi(\mathbf{x})^\top \phi(\mathbf{y})$ where $\phi(\mathbf{x}) = \mathbf{x}$ and $\phi(\mathbf{y}) = \mathbf{y}$.

(a) [10 points] $(\mathbf{x}^\top \mathbf{y})^2$

(b) [10 points] $(\mathbf{x}^\top \mathbf{y})^3$

(c) [20 points] $(\mathbf{x}^\top \mathbf{y})^k$ where $k$ is any positive integer.

| $x_1$ | $x_2$ | $x_3$ | $y$ |
|---|---|---|---|
| 1 | -1 | 2 | 1 |
| 1 | 1 | 3 | 4 |
| -1 | 1 | 0 | -1 |
| 1 | 2 | -4 | -2 |
| 3 | -1 | -1 | 0 |

Table 1: Linear regression training data.

# 2 Linear Regression

Suppose we have the training data shown in Table 1, from which we want to learn a linear regression model, parameterized by a weight vector $\mathbf{w}$ and a bias parameter $b$.

1. [10 points] Write down the LMS (least mean square) cost function $J(\mathbf{w}, b)$.
2. [30 points] Calculate the gradient $\frac{\nabla J}{\nabla \mathbf{w}}$ and $\frac{\nabla J}{\nabla b}$
   (a) when $\mathbf{w} = [0, 0, 0]^\top$ and $b = 0$;
   (b) when $\mathbf{w} = [-1, 1, -1]^\top$ and $b = -1$;
   (c) when $\mathbf{w} = [1/2, -1/2, 1/2]^\top$ and $b = 1$.
3. [20 points] What are the optimal $\mathbf{w}$ and $\mathbf{b}$ that minimize the cost function?
4. [50 points] Now, we want to use stochastic gradient descent to minimize $J(\mathbf{w}, b)$, we initialize $\mathbf{w} = \mathbf{0}$ and $b = 0$. We set the learning rate $r = 0.1$ and sequentially go through the 5 training examples. Please list the stochastic gradient in each step and the updated $\mathbf{w}$ and $b$.

# 3 Mistake Driven Learning Algorithm

Identify the maximum number of mistakes made by Halving algorithm in learning a target function from following concept classes. Please check whether the Halving algorithm is a mistake bound algorithm.

1. [10 points] Disjunction of $n$ boolean variables.
2. [10 points] Disjunction of $k$ boolean variables out of the total $n$ input variables. Note $k$ is a constant and smaller than $n$.
3. [10 points] $m$-of-$n$ rules. Note $m$ is a constant and smaller than $n$.
4. [20 points] All boolean function of $n$ input boolean variables.

# 4 Perceptron

1. Let us review the Mistake Bound Theorem discussed in our lecture.

   (a) [10 points] If we change the second assumption to be as follows: Suppose there exists a vector $\mathbf{u} \in \mathbb{R}^n$, and a positive $\gamma$, we have for each $(\mathbf{x}_i, y_i)$ in the training data, $y_i(\mathbf{u}^\top \mathbf{x}_i) \geq \gamma$. What is the upper bound for the number of mistakes made by the Perceptron algorithm? Note that $\mathbf{u}$ is unnecessary to be a unit vector.

   (b) [10 points] Following (a), if we do NOT assume $\mathbf{u}$ is a unit vector, and we still want to obtain the same upper bound introduced in the lecture, how should we change the inequalities in the second assumption?

   (c) [20 points] Now, let us state the second assumption in another way: Suppose there a hyperplane that can correctly separate all the positive examples from the negative examples in the data, and the margin for this hyper plane is $\gamma$. What is the upper bound for the number of mistakes made by Perceptron algorithm?

2. [20 points] We want to use Perceptron to learn a disjunction as follows,

   $$f(x_1, x_2, \ldots, x_n) = \neg x_1 \vee \neg \ldots \neg x_k \vee x_{k+1} \vee \ldots \vee x_{2k} \quad \text{(note that } 2k < n\text{)}.$$

   Please derive an upper bound of the number of mistakes made by Perceptron in learning this disjunction. Is Perceptron a mistake bound algorithm?

# 5 Programming Assignments

1. We will implement the LMS method for a linear regression task. The dataset is from UCI repository (`https://archive.ics.uci.edu/ml/datasets/Concrete+Slump+Test`). The task is to predict the real-valued SLUMP of the concrete, with 7 features. The features and output are listed in the file "regression/data-desc.txt". The training data are stored in the file "regression/train.csv", consisting of 53 examples. The test data are stored in "regression/test.csv", and comprise of 50 examples. In both the training and testing datasets, feature values and outputs are separated by commas.

   (a) [90 points] Implement the batch gradient descent algorithm, and tune the learning rate $r$ to ensure the algorithm converges. To examine convergence, you can watch the norm of the weight vector difference, $\|w_t - w_{t-1}\|$, at each step $t$. if $\|w_t - w_{t-1}\|$ is less than a tolerance level, say, $1e-6$, you can conclude that it converges. You can initialize your weight vector to be $\mathbf{0}$. Please find an appropriate $r$ such that the algorithm converges. To tune $r$, you can start with a relatively big value, say, $r = 1$, and then gradually decrease $r$, say $r = 0.5, 0.25, 0.125, \ldots$, until you see the convergence. Report the learned weight vector, and the learning rate $r$. Meanwhile, please record the cost function value of the training data at each step, and then draw a figure shows how the cost function changes along with steps. Use your final weight vector to calculate the cost function value of the test data.

(b) [90 points] Implement the stochastic gradient descent (SGD) algorithm. You can initialize your weight vector to be $\mathbf{0}$. Each step, you randomly sample a training example, and then calculate the stochastic gradient to update the weight vector. Tune the learning rate $r$ to ensure your SGD converges. To check convergence, you can calculate the cost function of the training data after each stochastic gradient update, and draw a figure showing how the cost function values vary along with the number of updates. At the beginning, your curve will oscillate a lot. However, with an appropriate $r$, as more and more updates are finished, you will see the cost function tends to converge. Please report the learned weight vector, and the learning rate you chose, and the cost function value of the test data with your learned weight vector.

(c) [20 points] We have discussed how to calculate the optimal weight vector with an analytical form. Please calculate the optimal weight vector in this way. Comparing with the weight vectors learned by batch gradient descent and stochastic gradient descent, what can you conclude? Why?

2. We will implement Perceptron for a binary classification task. The features and labels are listed in the file "classification/data-desc.txt". The training data are stored in the file "classification/train.csv", consisting of 872 examples. The test data are stored in "classification/test.csv", and comprise of 500 examples. In both the training and testing datasets, feature values and labels are separated by commas.

(a) [60 points] Implement the standard Perceptron. Set the maximum number of epochs $T$ to 10. Report your learned weight vector, and the average prediction error on the test dataset.

(b) [60 points] Implement the voted Perceptron. Set the maximum number of epochs $T$ to 10. Report the list of the distinct weight vectors and their counts — the number of correctly predicted training examples. Using this set of weight vectors to predict each test example. Report the average test error.

(c) [60 points] Implement the average Perceptron. Set the maximum number of epochs $T$ to 10. Report your learned weight vector. Comparing with the list of weight vectors from (b), what can you observe? Report the average prediction error on the test data.

(d) [20 points] Compare the average prediction errors for the three methods. What do you conclude?